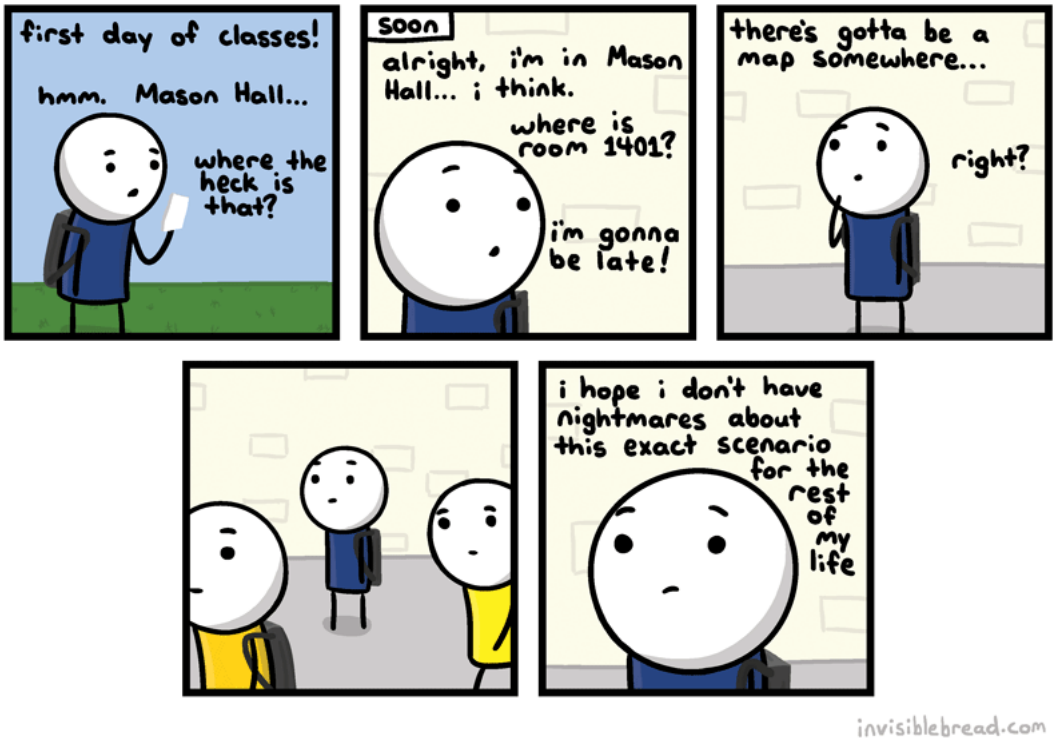
CS50 - Assignment 1
Getting started

Due: January 26th at 11:59pm



http://joyreactor.com/post/954711

For our first assignment, we will walk through the basics of Python and VS Code, play with Python a bit in this environment, and then make sure everything is set up to submit using Gradescope.

1 Assignment overview

For this first assignment we'll be doing a few different things:

- Install Python and VS Code.
- Create a VS Code project and a virtual environment.

- Play a bit with writing and running code in VS Code.
- Write our first program.
- Read and summarize our first ethics reading.
- Familiarize yourself with Gradescope for submitting assignments.

2 Python and VS Code

Most of the assignments in this class will use Python and VS Code.

- *If you are planning to use your own computer for this class*, you'll need to install Python and VS Code (see the instructions below). Don't wait until the last minute to do this in case you run into any problems.
- *If your are planning on using one of the computers in the CS labs* (Edmunds upstairs, 227 and 229), then you can skip the installation steps. You should be able to log into these computers with your normal Pomona login. Don't wait until the last minute to try and log in. If you do run into any issues, reach out to Corey LeBlanc in Edmunds 218 during regular business hours or the ITS helpdesk for other hours.
- *If you are also in CS51 this semester*, the instructions are more or less the same for installing Python and VS Code, so you only need to do it once.

2.1 Installing Python and VS Code on your own computer

You'll need to first install Python and then VS Code.

2.1.1 Installing Python

The latest Python interpreter can be downloaded from `https://www.python.org/downloads`. Once you visit this link, your operating system will be automatically detected. You should select the latest distribution of Python 3, Python 3.13.x, and install it after it is downloaded.

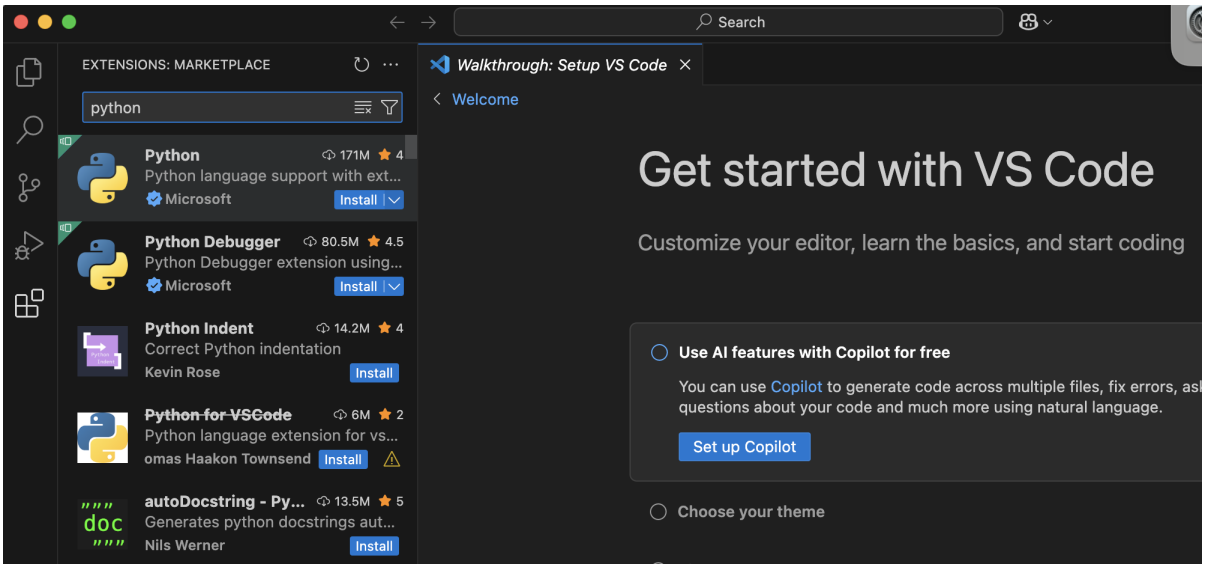
For Windows users, be sure to check the *Add to PATH* checkbox when installing Python!

2.2 Installing VS Code

Visit `https://code.visualstudio.com/download`. Your operating system won't be automatically detected. Install VS Code and then open it.

When you first run VS Code, you will be prompted with a number of options. *Make sure you do* ***NOT*** *set up Copilot* (i.e., do not set up "Use AI features with Copilot").

VS Code is a general-purpose IDE that can be used for a number of different programming languages. To use it for Python, you'll need to install the Python extension. Visit the extensions tab shown in the leftmost column and type `Python`. You want to install the Python extension package by Microsoft, as seen below.



Once this step is complete, you can confirm that Python (along with some other useful extensions that will come in handy later in the course) has been installed, and you can now write code in Python using VS Code.

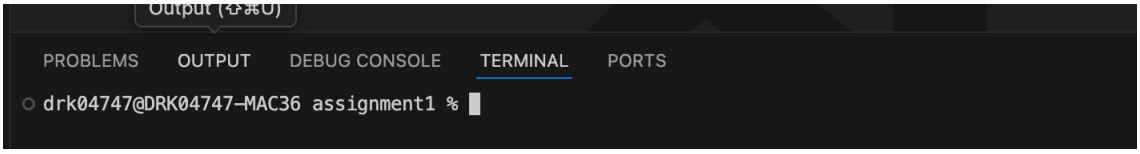
3 Using the Python interpreter

The most common way to interact with Python is to write code in a file and then run the file through Python. This is what we will do for a majority of the assignments in the class. However, Python can also be used interactively via the Python shell. When using the shell, you can type commands one at a time and the shell will "echo" the value that is returned from the command (if there is one). Playing with the shell is a great way to test out simple commands and we'll also use it to interactively test out our programs as we write them.

3.1 Terminal

The terminal is a program that provides a command-line interface (CLI), that is it uses text-based commands rather than a graphical user interface (GUI) to interact with the operating system. The terminal is extremely powerful and allows you to do everything that the GUI does, plus a lot more, despite a steep learning curve at the beginning. For this class, we'll mostly just use it to launch Python interactively.

Open VS Code. To use the terminal in VS Code, go to `View` and then `Terminal` or use the shortcut `Ctrl+‘` for Windows or `Control+‘` for macOS.



3.2 The Python shell

In your terminal, type `python3`. If you are on Windows and are facing issues in this step, jump to the Troubleshooting subsection and finish the steps there before coming back.

- Try a few simple mathematical equations (e.g. "1+1", "2**3", "(100/20)+45*7"). Notice that Python makes for a pretty easy to use calculator.
- 22/7 is an approximation for Pi. Type this in.
- Remember that we can use variables to store intermediary values. Assign 22/7 to a variable called `pi`. Use that variable to calculate the area of a circle with radius 15 (remember, the area of a circle is π times the radius squared).
- In 2011, Pomona built a new parking structure. For a variety of reasons, the college decided to put a field on top of the parking structure (pretty cool, right?!).

Since we're in southern California, we don't need any covering for this field. However, let's pretend that we anticipate bad weather more regularly and we decide we need to cover it with a dome (global warming, maybe?). If we make the approximating assumption that the field is a circle of radius 100 feet, what is the surface area of the dome that would cover it?

Hint: the surface area of a *sphere* is $4\pi r^2$.

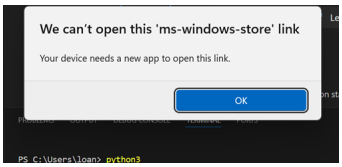
In the Python shell, you can use the up and down arrow keys to revisit commands you typed previously. Use the up arrow key to get your previous statement and then edit it to get the surface area of the dome assuming that the radius is 200 feet.

When you want to exit the Python shell, type `exit()`. If you have done something exceptionally time-consuming, e.g., you tried to calculate (`10**10**1000`) (don't!), chances are the Python shell or even your entire computer will slow down and might even crash. You can always hit `Ctrl/Command + C` to try to kill the process, but it's better to avoid such risky moves altogether.

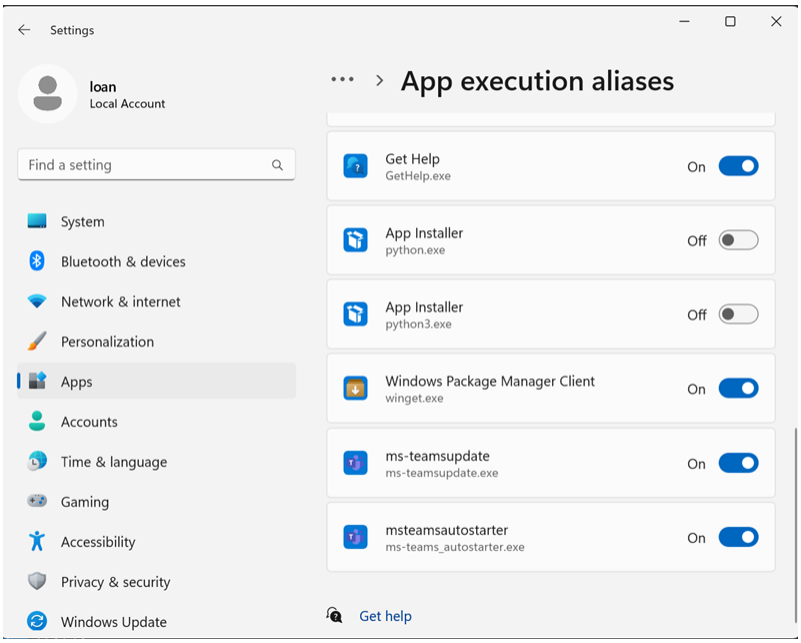
Note that there is nothing to explicitly submit for this section.

3.3 Troubleshooting working with the Python shell in Windows

If you are on Windows, there is a chance that you might get a pop-up window saying `We can't open this 'ms-window-store' link` when typing `python3`.

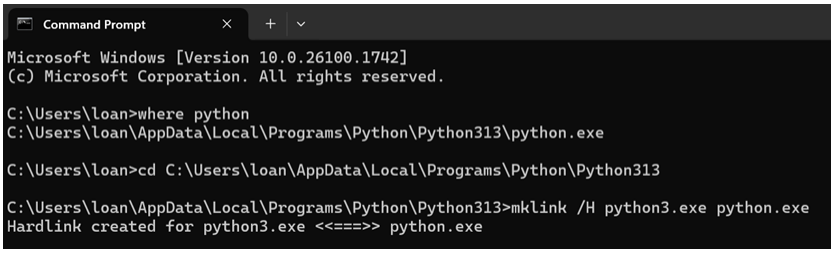


If that's the case, go to your search bar and type `App execution aliases`. You should uncheck the options `App Installer` for `python.exe` and `python3.exe`



You might still not be able to type `python3` and have it recognized. Go to your search bar and type `cmd`. The `Command Prompt` should launch.

Start by typing `where python`. This command will tell you where Python was installed. For example, for me it is in the directory `C:\Users\loan\AppData\Local\Programs\Python\Python313` but it might be a different directory for you. Copy the result you got (excluding the `python.exe`) and type `cd` and your Python path. Finally, type `mklink /H python3.exe python.exe`.



Close VS Code and relaunch it. You should be able to type `python3` now and have it recognized. If not, please reach out to the course staff.

4 Creating a Python project

Working with the Python shell is great if we want to use Python as a calculator and for some superficial work, but our needs as programmers quickly go beyond that. We will now see how to create a Python project and write and run code in files.

4.1 Creating a folder for your work

Python files are just normal text files with a `.py` extension that will reside on your computer like any other files. We'll create a folder for this class where you can store all of your assignments in one place and then we'll point VS Code at that.

- In your `Documents` folder, create a subfolder called `cs50`.
- In your new `cs50` folder, create a subfolder called `assignment1`.

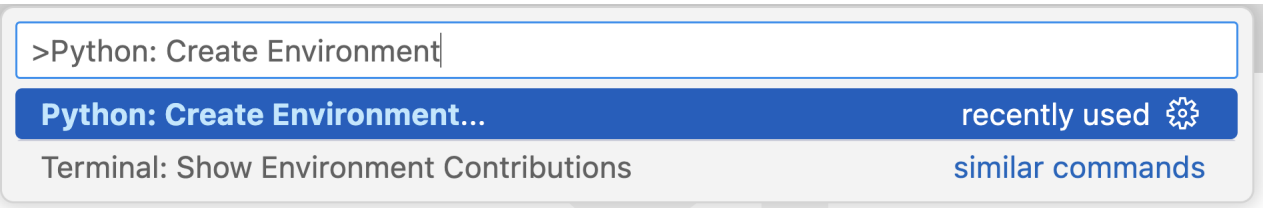
Each time you have an assignment, you'll make a new folder in your `cs50` folder for the work for that week.

4.2 Creating a virtual environment

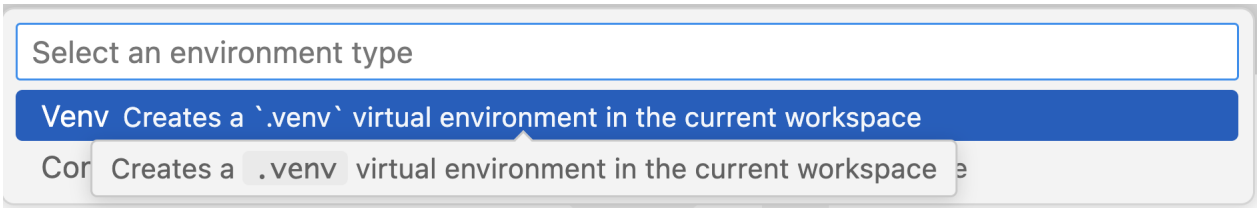
As we write more advanced Python code, we will find ourselves needing to install packages. To isolate changes to the specific projects we are working on, it is a good idea to create a *virtual environment*.

Go to `File>Add Folder to Workspace` and selecting the newly created `assignment1` folder under `Documents` and `cs50`. If prompted, agree that you trust the authors in this folder.

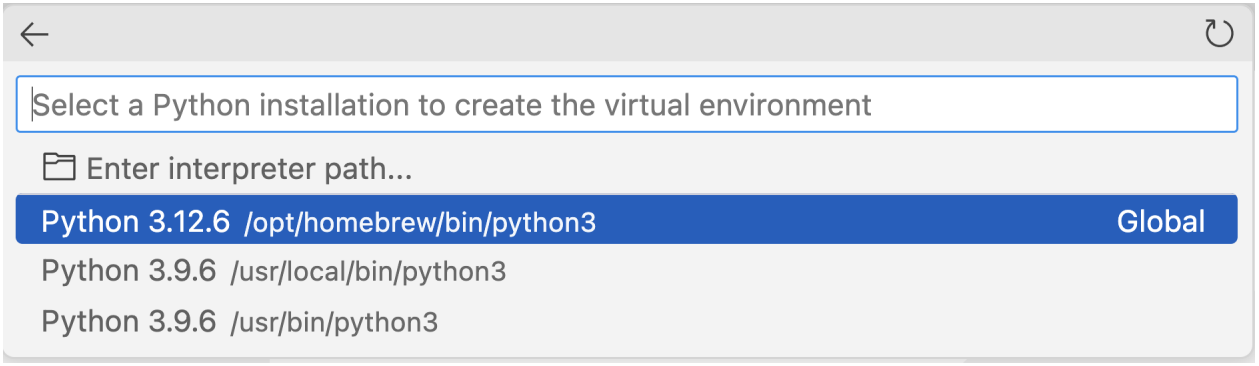
Bring up the `Command Palette` (on macOS: Command+Shift+P, on Windows: Ctrl+Shift+P) and type `Python: Create Environment`.



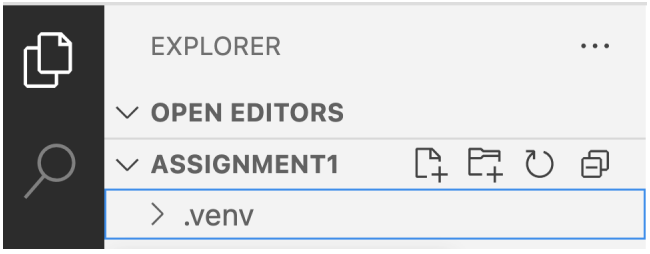
Select `venv`.



When prompted, pick the Python interpreter we installed. It should be something like `Python 3.13.`

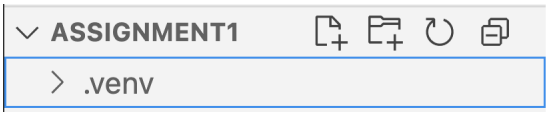


You can confirm that the virtual environment has been selected by looking in the Explorer (you'll see the `.venv` below the assignment folder):



4.3 Creating a new Python file

From the File Explorer toolbar, select the left-most `New File` button on the assignment1 folder (it looks like a page with a plus sign):



Name your file `assign1.py` (make sure you name it exactly this). The `.py` extension indicates that this file is not just plain text, but source code written in Python. It helps IDEs like VS Code highlight and make our Python code more readable.

Once you create the file, you will get a new empty file. Add a print statement to the top of the file:

```
print("This is my first Python program!!")
```

Select `Run Python File` from the top right:

In the Terminal window, you should see your text printed out.

Instead of using the VS Code `Run` button, you could have also typed on the terminal
`python3 assign1.py`
which is what VS Code runs behind the scenes.

If you wanted to run your code, but then be able to still interact with the Python sell, you could have also run it interactively using
`python3 -i assign1.py`
We'll see this interactive version can be helpful for debugging.

5 Types in Python

Everything in Python has a type. Run the Python shell again in the Terminal so that you can play with Python interactively. For each of the expressions below, see if you can figure out what the value will be (i.e., the result) as well as the type of that value will be. Then, check your answers using the shell. Remember, the `type` function will tell you the type of a value, e.g., you can type `type(3/2)` to figure out the type of the result of the expression.

1. `3/2`
2. `3//2`
3. `3.0/2`
4. `3.0//2`
5. `"3/2"`
6. `1.0 + 2`
7. `3.0 * 4 + 5`
8. `3 * (4 + 5)`
9. `3 / 2 + 1`
10. `"My favorite number is:  " + str(2**2**2 * 4 - 17)`

Put your answers as *comments* in your `assign1.py` with a comment above them stating that they are your type answers. For example, the beginning if your comments might look like:

```
# Type answers
# 1. 1 of type int
# 2. ...
```

6 Dr. Seuss in code

The following is the introduction to the book "Green Eggs and Ham" by Dr. Seuss:

```
I am Sam. I am Sam. Sam-I-Am.
That Sam-I-Am! That Sam-I-Am! I do not like that Sam-I-Am!
Do you like green eggs and ham?
I do not like them, Sam-I-Am.
I do not like green eggs and ham.
```

For the rest of the assignment, your goal is to write a program that prints out this introduction with the lowest "score". The score for your program will be the sum of the number of variables you use plus the total length (in characters, including spaces, punctuation, etc.) of all of the strings in your program. When your "run" (i.e., click the arrow in VS Code) your program should print out the above lyrics.

Restrictions: Variable declarations may not have any spaces in them *unless* they combine exactly two other variables with a space between them. For example, if you've defined

```
sam = "Sam"
am = "am"
```

then

```
am_sam = am + " " + sam
```

would be legal, but the following would **not** be legal:

```
am_sam_bad = "am Sam"
am_sam_bad2 = "am " + sam
```

The easiest way to recreate the first line would be:

```
print("I am Sam. I am Sam. Sam-I-Am.")
```

which would receive a score of 29, since it uses 0 variables and the string contains 29 characters.

Another option to recreate the first line would be:

```
sam = "Sam"

print("I am " + sam + ". I am " + sam + ". " + sam + "-I-Am.")
```

which would receive a score of 21. It uses 1 variable, plus strings totaling: $5 + 7 + 2 + 6 = 20$.

You will be graded based on:

- Correctness: whether your program correctly prints out the text
- Score: the lower the better
- Creativity/effort of your solution

There are *many* possible ways to solve this with varying scores. We do not expect you to get the optimal lowest score, but you should make some attempt to minimize the score. Make sure you use good variable names to help keep track of what is stored in them.

When you're happy with your solution, put the final score you think you achieved in a comment at the bottom of your program.

7 Comments

Commenting is an important of coding. We'll talk more about these and give examples as we get further into the class. For this first assignment, you should have:

– Comments at the very top of the file stating your name and the assignment number.

– Your commented solutions to the type question.

– A comment delimiting the Dr. Seuss solution from the type answers.

8 Ethics

As part of this course, we'll also be reading and discussing some of the ethical implications of programming and software development. Programming is a tool and like any tool, it can be used in a range of ways, some better than others. Read the following short article: `https://medium.com/@sterrkIT/the-importance-of-ethical-programming-4e7d09129dcd`

and then skim over Code of Conduct for the ACM (Association for Computing Machinery):

`https://www.acm.org/code-of-ethics`

In a text editor (pick whatever you're comfortable with), write 1–2 paragraphs and discuss anything that surprised you and any questions you might have. List at least one example of a real-word softward that is questionable ethically. Export your document as a pdf, called `ethics1.pdf`. I recommend saving this file in your `cs50/assignment1/` folder (which should also have your `assign1.py` file).

9 When you're done

For this assignment, you will be submitting two files: `assign1.py` and `ethics1.pdf`. We will be submitting all assignments through https://www.gradescope.com/. The grading and feedback will also be returned through Gradescope as well. In addition, you will be able to submit regrade requests through Gradescope. You should have received an email notification that you were added to the Gradescope site with directions on how to login. Please let us know ASAP if that is not the case.

When you log into Gradescope, you should see CS50 on your dashboard. If you click on the course, you'll see the list of assignments that you have submitted and work that is due in the near term (for now, just Assignment 1).

To submit the assignment, click on the assignment name and a window will pop up for you to submit the code, via "Drag and drop" or "Click to browse", as shown in the following image.

Some assignments (like this one) may require you to upload multiple files. In this case, you should submit all the required files together at once, rather than submitting them one by one. With "Drag and drop", you can select multiple files to perform the operation. With "Click to browse", you can select multiple files by pressing "Command" and clicking on multiple files to select them for uploading.

Remember, your files should all be in `Documents/cs50/assignment1/`.

Note that you can resubmit your work many times as long as it is before the deadline. The very last submission will be considered for grading.

10 Grading

	points
Types in python	5
Dr. Seuss in code	5
Correct submission	1
Comments	2
Ethics prompt	1
Total	14